Techniques de Flots et Comparaison d'Images en 2D

Moncef DAOUD[(1)] **- Ali Ridha MAHJOUB**[(2)] **- Mikael TANGUY**[(2)]

SPO, Equipe d'Informatique de l'UBO

[(1)]Université de Bretagne Sud
4, rue Jean Zay, 56325 Lorient - France.
e-mail: daoud@univ-ubs.fr

[(2)]Université de Bretagne Occidentale
6, av. le Gorgeu, BP 809, 29285 Brest - France.
e-mail: mahjoub@univ-brest.fr

RÉSUMÉ

Récemment, une approche efficace utilisant des techniques de flots a été développée pour des problèmes de traitement d'images dans le cas 1D. Nous étudions dans ce papier une extension de cette approche dans le cas 2D. Nous présentons un algorithme de décomposition du graphe associé aux images. Opérant sur la configuration réelle de l'image, cet algorithme permet d'aboutir à la solution optimale en deux procédures (la détermination d'une solution réalisable et l'optimisation des cycles).

ABSTRACT

Recently, an efficient approach has been developped for image processing problems in the case 1D, using techniques from flow theory. In this paper we study an extension of that approach to the case 2D. We give a decomposition algorithm for the graph associated to the images. Using the real configuration, this algorithm permits to provide an optimal solution using two procedures (the determination of a feasible solution and cycle optimization).

1 Introduction

Dans la littérature, on trouve diverses techniques de comparaison et d'approximation d'images basées sur la mesure de disparité entre deux images. Une de ces techniques est celle de Hutchinson [4] qui utilise une distance développée à l'origine pour des problèmes de comparaison et d'approximation d'images fractales. Le problème de comparaison de deux images se ramène au calcul de la distance entre les deux mesures de probabilités modélisant les images. F. Barahona et al [1] ont montré que le calcul de cette distance peut se formuler comme un problème de flots. Ils ont développé un algorithme efficace qui permette de calculer cette distance dans le cas 1D [2]. Le but de cette étude est d'étendre cette approche au cas 2D.

2 Distance de Hutchinson et le problème de flots

La comparaison de deux images, modélisées par deux mesures de probabilités, μ et ν, se ramène au calcul de la distance de Hutchinson [4]. Dans le cas discret 1D avec n éléments, la distance de Hutchinson est définie par la formule suivante :

$$d_H(\mu,\nu) = \sup_f\{\sum_i f_i.(\mu_i - \nu_i)\}$$

$$\text{sc} \begin{cases} \mid f_i - f_{i+1} \mid \leqslant 1;\ i = 1,\cdots,n-1 \\ f_1 = 0,\ f \text{ lipschitzienne de classe } C^1 \end{cases}$$

Barahona et al [2] ont montré que ce type de problème est équivalent à un programme linéaire et dont le dual correspond à un problème de flots et ce pour toute dimension. Par exemple, dans le cas 1D, le problème dual est défini par :

$$\min \sum_{i=1}^{n-1}(x_{i,i+1} + x_{i+1,i})$$

$$\text{sc} \begin{cases} x_{1,2} - x_{2,1} = \mu_1 - \nu_1, \\ x_{n,n-1} - x_{n-1,n} = \mu_n - \nu_n, \\ x_{i,i+1} + x_{i,i-1} - x_{i-1,i} - x_{i+1,i} = \mu_i - \nu_i, \\ \text{pour i} = 2,\cdots,\text{n} - 1 \end{cases}$$

où $x_{i,j}$ représente la valeur du flot entre les nœuds i et j : Pour

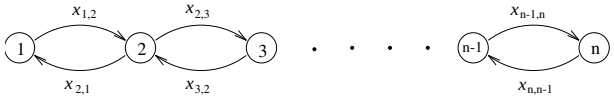

FIG. 1 — Structure du graphe 1D

résoudre ce problème, Barahona et al [2] ont développé un algorithme linéaire, opérant par décomposition du graphe de base en graphe de flots. Nous en présentons une illustration.

Soient les deux distributions de probabilités suivantes $\mu = \{a_i\}$ et $\nu = \{b_i\}$ modélisant deux images à une dimension, A et B :

$$\begin{cases} \mu = (0.2\ \ 0.1\ \ 0.3\ \ 0.1\ \ 0.3) \\ \nu = (0.0\ \ 0.6\ \ 0.2\ \ 0.2\ \ 0.0) \end{cases}$$

Le graphe associé à ces deux images est le suivant : L'exécution de l'algorithme de Barahona et al [2] sur le graphe

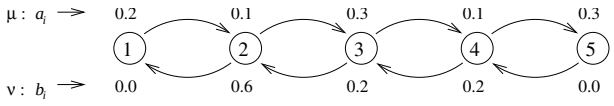

FIG. 2 — Graphe des distributions de probabilités

1D nous donne la décomposition en chemins de flots suivante : La distance de Hutchinson est déterminée par la somme des

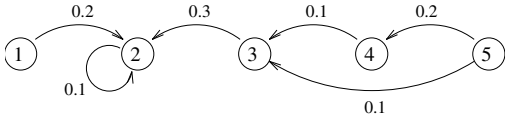

FIG. 3 — Graphe des chemins de flots

produits des flots avec les longueurs affectées à chaque chemin :

$$d_H(\mu, \nu) = \sum_{q=1}^{n} l_q \cdot f_q$$

Cette approche nous a incités à développer un algorithme de décomposition de graphe 2D qui tienne compte de la configuration réelle de l'image. Cet algorithme fait appel à une procédure permettant d'optimiser un cycle, que nous avons élaborée, en nous basant sur l'algorithme de Barahona et al [2]. Dans ce qui suit, nous présentons cet algorithme.

3 Optimisation d'un cycle

Dans un premier temps, nous avons étendu l'algorithme de Barahona et al [2] au cas d'un cycle. En effet, en cassant le cycle on obtient un graphe 1D. C'est cette idée qui a été exploitée. Notre algorithme d'optimisation d'un cycle est le suivant :

1. casser le cycle entre deux nœuds voisins (avec un choix arbritraire des nœuds),
2. appliquer l'algorithme de Barahona [2] sur le graphe 1D obtenu à l'étape 1,
3. ramener le graphe en flots de longueur 1, en décomposant les arcs de longueur supérieure à 1 en une séquence d'arcs de longueur 1,
4. déterminer
   - n : nombre de sommets du cycle
   - S_1 : sens majoritaire des flots obtenus à l'étape 3.
   - S_2 : sens inverse de S_1
   - N_1 : nombre de flots dans le sens S_1
   - $flot_{min}$: minimum des flots envoyés dans le sens S_1
5. si N_1 est supérieur à la partie entière de $\frac{n}{2}$ alors : envoyer un flot de valeur $flot_{min}$ sur tout le cycle dans le sens S_2 et retour à l'étape 4.
6. calculer la distance de Hutchinson recherchée : $\sum_{i=1}^{n} flot_{i,i\oplus 1}$[1]

Dans ce qui suit, nous présentons un exemple d'optimisation d'un cycle à cinq nœuds :

[1] Si les sommets sont numérotés de 1 à n alors $i \oplus 1 = i \bmod n + 1$

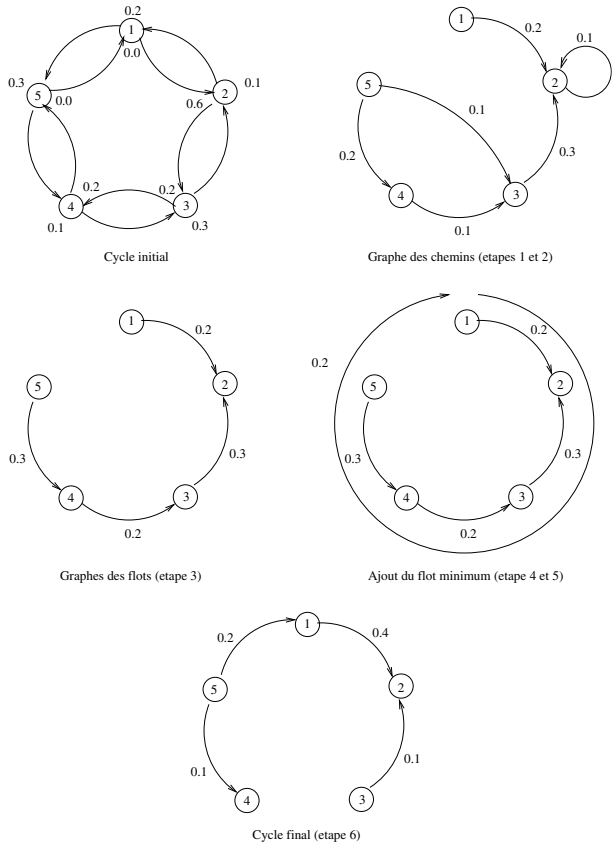

FIG. 4 — Exemple de l'algorithme d'optimalité

4 Algorithme de décomposition du graphe 2D par les flots

Pour le cas 2D, les images à comparer sont modélisées par deux distributions de probabilités. Ces distributions sont ramenées à des valeurs entières en les multipliant par un facteur approprié (PPCM), afin d'éviter les problèmes de calcul flottant. Par la suite, on élabore le graphe 2D de retenues de flots (différence des distributions de probabilités) qui servira comme graphe de départ pour l'algorithme de décomposition.

Nous présentons un exemple de modélisation sur un modèle réduit de deux images binaires 3x3 : Pour chaque image

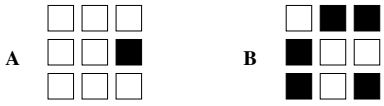

FIG. 5 — Images de départ

binaire ou à niveaux de gris, on multiplie le niveau de gris de chaque pixel par le PPCM des deux sommes de niveaux de gris de chacune des deux images. Ensuite, on divise par la somme des niveaux de gris de l'image associée.

Les sommes de niveaux de gris, obtenues avec les images A et B, sont respectivement 8 et 4, donc le facteur PPCM est

8.

$$\begin{array}{ccc|ccc|c|ccc|ccc} & A & & & B & & & & \mu & & & \nu & \\ 1 & 1 & 1 & 1 & 0 & 0 & \text{PPCM} & 1 & 1 & 1 & 2 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & \Longrightarrow & 1 & 1 & 0 & 0 & 2 & 2 \\ 1 & 1 & 1 & 0 & 1 & 0 & & 1 & 1 & 1 & 0 & 2 & 0 \end{array}$$

Le graphe de modélisation 2D, différence des deux distributions de probabilités entières se présente comme suit : L'algorithme de décomposition du graphe 2D doit respecter

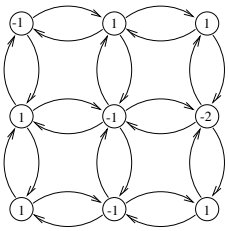

FIG. 6 — Graphe de modélisation

la notion de voisinage, où chaque nœud dépend de ses quatre voisins. Pour cela, nous proposons la décomposition de proche en proche, nœud par nœud. Le traitement d'un nœud consiste à envoyer les retenues de flots de ce nœud sur les nœuds voisins, en se limitant au maillage élémentaire carré à quatre sommets (graphe temporaire). Ainsi, on aura à la fin de la procédure au plus un arc non nul, entre deux nœuds, représentant le flot transféré d'un nœud à un autre. La distance de Hutchinson, mesurant la disparité des deux images, sera la somme des flots affectés aux arcs.

Nous détaillons, dans la suite, la décomposition du graphe temporaire 2x2 en quatre phases :

Phase 1 :

On calcule les retenues de flots du graphe temporaire de la façon suivante :

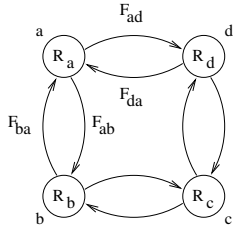

FIG. 7 — Structure du graphe temporaire

R_a = retenue de flot du nœud traité,
R_b = somme des retenues de flots de tous les nœuds des lignes situées en-dessous de la ligne du nœud traité,
R_d = somme des retenues de flots de tous les nœuds des colonnes situées à droite de la colonne du nœud traité,
R_c = complément à zéro de la somme $R_a + R_b + R_d$.

Phase 2 :

Le calcul des flots F_{ad} (flot allant du nœud a vers le nœud d) et F_{ba} (flot allant du nœud b vers le nœud a) est basé sur les formules suivantes :

$$F_{ad} = \frac{R_a + R_b - R_d}{2},\ F_{ba} = \frac{R_b - R_d - R_a}{2}$$

Phase 3 :

Pour optimiser davantage la solution, on minimise les flots affectés aux arcs (a, d) et (b, a) selon le principe suivant :

si $F_{ba} \times F_{ad} > 0$ **alors**
  si $F_{ba} > 0$ **alors** $F = \min(F_{ad}, F_{ba})$
  sinon $F = \max(F_{ad}, F_{ba})$
  finsi
  $F_{ad} = F_{ad} - F$; $F_{ba} = F_{ba} - F$
finsi

Phase 4 :

La mise à jour des retenues de flots dans le graphe pour un nœud (i, j) se fait comme suit :

$$R_{i,j} = 0,\ R_{i,j+1} = R_{i,j+1} + F_{ad},\ R_{i+1,j} = R_{i+1,j} - F_{ba}$$

L'exemple suivant montre l'algorithme de décomposition sur le modèle réduit 3x3 de la Figure 6. On commence par traiter le nœud (1,1), en décomposant le graphe élémentaire composé des nœuds (1,1), (1,2), (2,1), (2,2), avec l'annulation de la retenue de flot du nœud (1,1) et l'envoi de l'excès de flot aux nœuds voisins (1,2) et (2,1).

Le traitement du nœud $(1, 1)$ nous donne, à la suite des phases 1 et 2, les résultats suivants :

$$\begin{aligned} R_a &= R_{1,1} = -1 \\ R_b &= R_{2,1} + R_{2,2} + R_{2,3} + R_{3,1} + R_{3,2} + R_{3,3} = -1 \\ R_d &= R_{1,2} + R_{1,3} + R_{2,2} + R_{2,3} + R_{3,2} + R_{3,3} = -1 \\ R_c &= -(R_a + R_b + R_d) = 3 \\ F_{ad} &= \tfrac{-1-1+1}{2} = -0.5,\ F_{ba} = \tfrac{-1+1+1}{2} = 0.5 \end{aligned}$$

Dans cet exemple, la phase 3 d'optimisation ne change pas les valeurs des flots F_{ad} et F_{ba}. L'intégration de ce graphe dans le

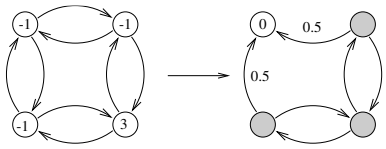

FIG. 8 — Décomposition du sous-graphe

graphe de départ nous donne le résultat de l'itération 1 de la Figure 9.

On continue le traitement selon le principe précédent, nœud par nœud et ligne par ligne, à l'exception de la dernière ligne et de la dernière colonne où l'excès de flots de chaque nœud est à renvoyer respectivement au nœud de droite et au nœud du dessous.

La solution finale obtenue par l'application de l'algorithme de décomposition du graphe 2D est généralement réalisable mais pas optimale, à cause de la présence de cycles non optimaux (au sens de l'algorithme cyclique). La détection de ces cycles suivie de l'application de l'algorithme cyclique d'optimalité permettra d'améliorer la solution réalisable, par étapes successives, et d'aboutir à la solution optimale.

5 Amélioration de la solution

Pour optimiser la solution issue de l'algorithme de décomposition en 2D de la section précédente, on itère le processus

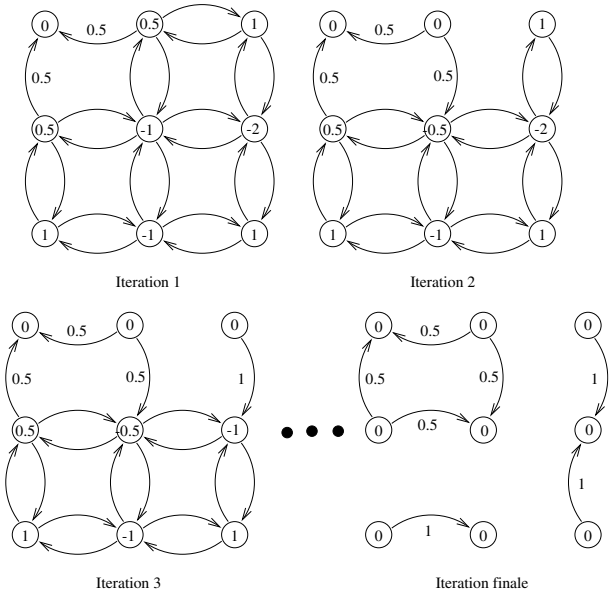

FIG. 9 — Exemple de décomposition

de recherche d'un cycle non optimal (au sens de l'algorithme cyclique) suivi de l'algorithme d'optimalité jusqu'à l'épuisement de cycles non optimaux.

Nous nous sommes intéressés, dans un premier temps, à rechercher les cycles non optimaux ayant une configuration bien particulière, par exemple, les cycles rectangulaires 2x3. Après optimisation de ces cycles, la solution obtenue n'est toujours pas optimale, à cause de la présence de certains cycles ayant une configuration plus complexe. Ainsi, nous avons opté pour la généralisation de la recherche de ces cycles.

Cette étape de recherche de cycles non optimaux est en cours de développement.

6 Résultats

Les tests ont été réalisés sur une station Sparc 5, avec une séquence d'images synthétiques binaires de taille 25x25. Les résultats de comparaison de chacune des images avec l'image source traduisent parfaitement la disparité croissante entre les deux images selon la distance de Hutchinson calculée avec cette technique de flots pour le cas 2D.

La comparaison des résultats obtenus avec des algorithmes existants de minimisation d'un problème de flot, comme celui de A. Goldberg [3], a montré l'efficacité de la méthode et sa performance, malgré la complexité du graphe de départ. La différence légère des résultats s'explique par le fait que la phase de recherche des cycles n'est pas encore généralisée.

7 Conclusion

L'originalité de ce travail réside dans l'extension du calcul de la distance de Hutchinson au cas 2D, au moyen des outils de la théorie des flots. Pour cela, nous avons mis en œuvre un algorithme de décomposition, s'appliquant sur le maillage du

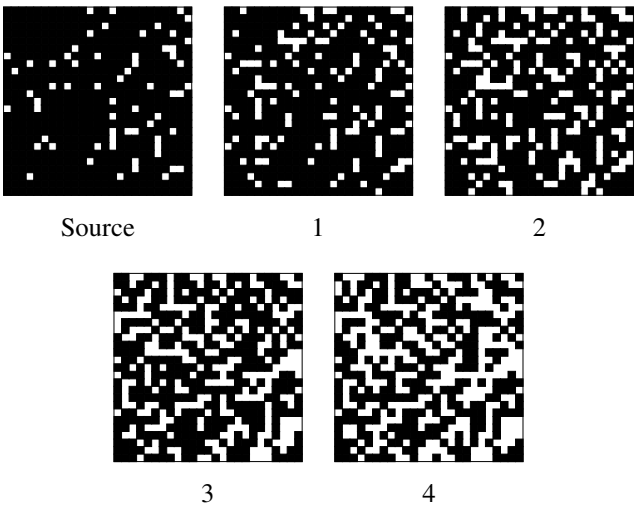

Image cible	distance Hutchinson		temps (s)	
	Algo1	Algo2	Algo1	Algo2
1	0.4322	0.3418	0.5500	1.1833
2	0.5515	0.4390	0.5500	1.1333
3	0.7972	0.6015	0.6167	1.1167
4	0.8889	0.7102	0.6167	0.9667

Algo1 : algorithme de décomposition
Algo2 : algorithme de flot de coût minimum [3]

graphe 2D, qui permet de retrouver une solution réalisable au problème de flots. Sachant que cette solution n'est pas en général optimale, un outil utilisant conjointement un algorithme de détection de cycles non optimaux et l'algorithme cyclique d'optimalité donne la solution optimale.

Dans la perspective de valider cet algorithme, des applications concrètes devront être réalisées, telles l'imagerie médicale, la robotique, la reconnaissance des formes.

Références

[1] F. Barahona, C.A. Cabrelli, U.M. Molter : "Computing the Hutchinson Distance By Networks Flow Problem", Random and Computational Dynamics, 1(1) : 117-119, 1992.

[2] F. Barahona, C.A. Cabrelli, U.M. Molter : "Matching Probability Measures On The Line Under Translation", Random and Computational Dynamics, 3 :121-135, 1995.

[3] A. Goldberg : "An efficient Implementation of a Scaling Minimum-Cost Flow Algorithm", Technical Report STAN-CS-92-1439, Computer Science Departement, Stanford University, Stanford, CA, 1992.

[4] J.E. Hutchinson : "Fractals and Self Similarity ", Indiana University Journal of Mathematics, 30 : 713-747, 1981.